

Laporan Proyek Data Science Menggunakan Metodologi CRISP-DM

A. Business Understanding (Pemahaman Bisnis)

Pada tahap awal ini, fokus utama adalah memahami permasalahan yang ingin diselesaikan. Proyek ini bertujuan untuk **memprediksi jenis wilayah (Perkotaan atau Perdesaan)** berdasarkan beberapa indikator pendidikan yang tersedia dalam dataset *Rapor Publik Asesmen Nasional*.

Tujuan Proyek

- Mengembangkan model machine learning yang mampu mengklasifikasikan jenis wilayah sekolah.
- Membantu pemangku kebijakan atau peneliti dalam melakukan analisis kondisi pendidikan berdasarkan karakteristik wilayah.

B. Data Understanding (Pemahaman Data)

Dataset yang digunakan bernama **raporPublikAsesmenNasional.csv**, yang berisi informasi mengenai beberapa indikator hasil Asesmen Nasional.

Sumber Data: <https://data.kemendikdasmen.go.id/dataset/p/asesmen-nasional/rapor-publik-asesmen-nasional-2024-kepala-satuan-pendidikan-2024-indonesia>

Isi Dataset (Contoh Kolom)

- Npsn
- nama_satuan_pendidikan
- kabupaten_kota
- provinsi
- status_sekolah
- kategori_wilayah (target)
- Indikator nilai seperti literasi, numerasi, dan karakter lainnya.

C. Data Preparation (Persiapan Data)

1. Menghapus data duplikat

Beberapa baris data yang identik dihapus agar tidak memengaruhi hasil training.

2. Menangani nilai kosong

Untuk kolom numerik → diisi dengan median atau mean. Untuk kolom kategorikal → diisi kategori terbanyak (mode).

3. Menangani outlier

Memeriksa fitur jumlah siswa, jumlah guru, rasio, dan fitur-fitur indikator lain untuk memastikan tidak ada nilai ekstrem yang merusak model.

4. Encoding data kategorikal

Mengubah kategori menjadi angka menggunakan:

- Label Encoding
- One Hot Encoding (jika diperlukan)

Contoh yang di-encode: kurikulum, jenis_sek, status sekolah, internet/listrik.

5. Normalisasi (opsional)

Karena Random Forest tidak terlalu sensitif terhadap skala, normalisasi hanya diterapkan jika ada fitur tertentu yang perbedaannya terlalu besar.

D. Modeling (Pemodelan)

Algoritma yang digunakan adalah Random Forest Classifier, karena cocok untuk dataset dengan banyak fitur numerik dan kategorikal, serta menghasilkan performa yang stabil.

Langkah-langkah pemodelan:

1. Membagi data menjadi training dan testing (80:20).
2. Melatih model Random Forest menggunakan data training.
3. Mengatur parameter seperti jumlah tree, depth, dan criterion jika diperlukan.
4. Melakukan prediksi pada data testing.

5. Menyimpan model terbaik untuk evaluasi.

Alasan pemilihan Random Forest:

1. Tahan terhadap noise dan missing value kecil
2. Tidak membutuhkan scaling
3. Bisa menangani fitur numerik dan kategorikal
4. Menghasilkan feature importance untuk melihat faktor paling berpengaruh
5. Umumnya akurat di dataset pendidikan

E. Evaluation (Evaluasi)

Pada tahap evaluasi, performa model diuji menggunakan tiga metode utama, yaitu Accuracy Score, Confusion Matrix, dan Classification Report.

- **Accuracy Score** digunakan untuk melihat seberapa besar persentase prediksi model yang benar.
- **Confusion Matrix** digunakan untuk mengetahui distribusi prediksi benar dan salah pada tiap kelas (Urban dan Rural).
- Sedangkan **Classification Report** memberikan informasi lebih detail melalui nilai Precision, Recall, dan F1-Score sehingga dapat terlihat apakah model bekerja seimbang untuk kedua kelas.

Hasil evaluasi dari ketiga metode ini membantu memastikan bahwa model Random Forest dapat memprediksi jenis wilayah sekolah secara akurat dan tidak bias ke salah satu kelas.

F. Deployment (Penerapan / Implementasi)

Model akan di-deploy menggunakan Gradio sehingga dapat digunakan dengan mudah.

Desain Interface Gradio:

Bagian Input (yang diisi oleh pengguna)

Ini semacam form berisi kolom-kolom yang bisa diisi, misalnya:

- jumlah peserta didik
- jumlah pendidik
- proporsi guru S1

- kurikulum
- status sekolah
- indikator mutu (misal APK, BBS, COP, dll)

Pengguna akan memasukkan angka atau memilih kategori di setiap kolom.

Tombol “Prediksi”

Setelah data diisi, pengguna menekan tombol Submit / Predict.

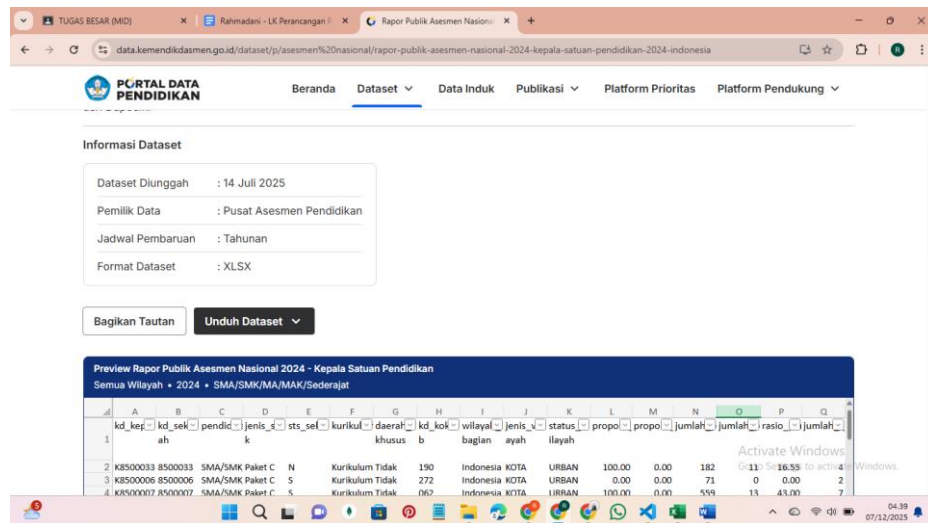
Bagian Output (hasil dari model)

Setelah tombol ditekan, model akan menampilkan:

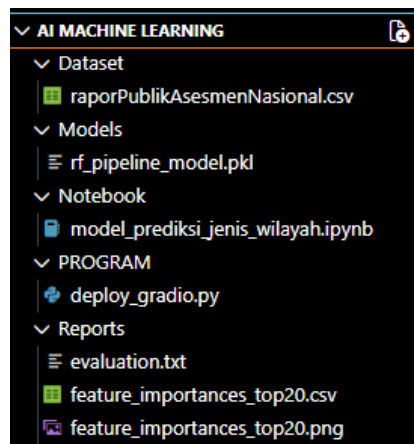
- Prediksi wilayah sekolah → Urban / Rural
- Probabilitas prediksi
- Fitur yang paling berpengaruh terhadap prediksi (feature importance)

LAMPIRAN

1. Sumber Dataset



2. Struktur File & Folder



3. Notebook/model_prediksi_jenis_wilayah.ipynb

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import os

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder, LabelEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.ensemble import RandomForestClassifier
```

```
DATA_PATH = "../Dataset/raporPublikAsesmenNasional.csv"

print("Dataset exists:", os.path.exists(DATA_PATH))
```

✓ 0.0s

Dataset exists: True

```
df = pd.read_csv(DATA_PATH, sep=';')
df.head()
```

	kd_kepsek_an	kd_sekolah	pendidikan_sederajat	jenis_sek	sts_sek	kurikulum	daerah_khusus	kd_kokab	wilayah_bagian	jenis_wilayah	SMK_KPL	SMK_PBEL	SMK_PIM	SMK_PKL	SMK_PKUR	SMK-SARPRAS	SMK_TIFA	SSV	TOC	TOR
0	KB500033	8500033	SMA/SMK/MA	Paket C	N	Kurikulum Merdeka	Tidak	190	Indonesia Barat	KOTA	—	NaN	NaN	NaN	NaN	NaN	NaN	57,97	36,05	53,50
1	KB500006	8500006	SMA/SMK/MA	Paket C	S	Kurikulum 2013	Tidak	272	Indonesia Barat	KOTA	—	NaN	NaN	NaN	NaN	NaN	NaN	52,20	35,16	62,48
2	KB500007	8500007	SMA/SMK/MA	Paket C	S	Kurikulum Merdeka	Tidak	62	Indonesia Barat	KOTA	—	NaN	NaN	NaN	NaN	NaN	NaN	61,35	36,05	60,14
3	KB500008	8500008	SMA/SMK/MA	Paket C	S	Kurikulum Merdeka	Tidak	104	Indonesia Barat	KABUPATEN	—	NaN	NaN	NaN	NaN	NaN	NaN	75,98	33,18	37,70
4	KB500005	8500005	SMA/SMK/MA	Paket C	S	Kurikulum Merdeka	Tidak	59	Indonesia Barat	KABUPATEN	—	NaN	NaN	NaN	NaN	NaN	NaN	47,90	30,85	43,90

5 rows x 79 columns

```
print(df.shape)
df.info()
df.describe()
```

```
(1478, 79)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1478 entries, 0 to 1477
Data columns (total 79 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   kd_kepsek_an                          1478 non-null   object
1   kd_sekolah                            1478 non-null   int64
2   pendidikan_sederajat                  1478 non-null   object
3   jenis_sek                             1478 non-null   object
4   sts_sek                               1478 non-null   object
5   kurikulum                             1364 non-null   object
6   daerah_khusus                         1478 non-null   object
7   kd_kokab                              1478 non-null   int64
8   wilayah_bagian                        1478 non-null   object
9   jenis_wilayah                         1478 non-null   object
10  status_wilayah                        1478 non-null   object
11  proporsi_pendidik_min_s1              1364 non-null   object
12  proporsi_pendidik_sertifikasi          1364 non-null   object
13  jumlah_peserta_didik                  1364 non-null   float64
14  jumlah_pendidik                       1364 non-null   float64
15  rasio_pendidik_peserta_didik           1364 non-null   object
16  jumlah_r_kelas                         1364 non-null   float64
17  ketersediaan_internet                  1364 non-null   object
18  ketersediaan_listrik                   1364 non-null   object
...
77  TOC                                    1096 non-null   object
78  TOR                                    1091 non-null   object
dtypes: float64(8), int64(2), object(69)
memory usage: 912.3+ KB
```

	kod_sekolah	kod_kokab	jumlah_peserta_didik	jumlah_pendidik	jumlah_r_kelas	jumlah_komp_milik	jumlah_perpus	jumlah_rombel	jumlah_siswa_rombel	jumlah_siswa_penerima_PIP
count	1.478000e+03	1478.000000	1364.000000	1364.000000	1364.000000	1364.000000	1364.000000	1364.000000	1364.000000	1364.000000
mean	6.378087e+06	252.335589	379.151760	26.032991	14.236804	6.425953	0.999267	13.794721	123.730205	116.871701
std	1.044141e+06	150.943737	387.046502	21.566623	10.545799	5.877043	0.514642	11.096870	129.599090	130.194433
min	5.100001e+06	1.000000	8.000000	0.000000	1.000000	0.000000	0.000000	2.000000	2.000000	0.000000
25%	5.100377e+06	123.000000	110.000000	11.000000	6.000000	3.000000	1.000000	6.000000	34.000000	30.000000
50%	7.100050e+06	245.000000	229.500000	19.000000	11.000000	5.000000	1.000000	10.000000	74.000000	67.000000
75%	7.100247e+06	379.750000	524.500000	35.000000	19.000000	8.000000	1.000000	18.000000	173.000000	158.000000
max	9.500001e+06	513.000000	2523.000000	138.000000	84.000000	58.000000	5.000000	84.000000	823.000000	801.000000

```
target = "jenis_wilayah"
df[target].value_counts()
```

✓ 0.0s

```
jenis_wilayah
KABUPATEN    1154
KOTA          324
Name: count, dtype: int64
```

```
numeric_cols = df.select_dtypes(include=['int64', 'float64']).columns.tolist()
categorical_cols = df.select_dtypes(include=['object']).columns.tolist()

# Hapus target dari categorical
if target in categorical_cols:
    categorical_cols.remove(target)

numeric_cols, categorical_cols
```

```
(['kd_sekolah',
  'kd_kokab',
  'jumlah_peserta_didik',
  'jumlah_pendidik',
  'jumlah_r_kelas',
  'jumlah_komp_milik',
  'jumlah_perpus',
  'jumlah_rombel',
  'jumlah_siswa_rombel',
  'jumlah_siswa_penerima_PIP'],
 ['kd_kepsek_an',
  'pendidikan_sederajat',
  'jenis_sek',
  'sts_sek',
  'kurikulum',
  'daerah_khusus',
  'wilayah_bagian',
  'status_wilayah',
  'proporsi_pendidik_min_si',
  'proporsi_pendidik_sertifikasi',
  'rasio_pendidik_peserta_didik',
  'ketersediaan_internet',
  'ketersediaan_listrik',
  'rasio_siswa_penerima_PIP',
  'SES_sekolah',
  ...
  'SMK-SARPRAS',
  'SMK_Tefa',
  'SSV',
  'TOC',
  'TOR'])
```

```
df = df.drop_duplicates()

for col in numeric_cols:
    df[col] = df[col].fillna(df[col].median())

for col in categorical_cols:
    df[col] = df[col].fillna(df[col].mode()[0])
```

```
le = LabelEncoder()
df[target] = le.fit_transform(df[target])
le.classes_
```

✓ 0.0s

```
array(['KABUPATEN', 'KOTA'], dtype=object)
```

```
X = df[numeric_cols + categorical_cols]
y = df[target]

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y)
```



```

preprocess = ColumnTransformer([
    ("categorical", OneHotEncoder(handle_unknown="ignore"), categorical_cols)
], remainder="passthrough")

model = RandomForestClassifier(
    n_estimators=200,
    max_depth=None,
    random_state=42
)

pipeline = Pipeline([
    ("preprocess", preprocess),
    ("model", model)
])

```

```

pipeline.fit(X_train, y_train)
print("Model Training Selesai!")

```

✓ 3.5s

Model Training Selesai!

```

y_pred = pipeline.predict(X_test)

acc = accuracy_score(y_test, y_pred)
cls = classification_report(y_test, y_pred)
cm = confusion_matrix(y_test, y_pred)

print("Accuracy:", acc)
print("\nClassification Report:\n", cls)
print("\nConfusion Matrix:\n", cm)

```

Accuracy: 0.8040540540540541

Classification Report:

	precision	recall	f1-score	support
0	0.80	1.00	0.89	231
1	0.89	0.12	0.22	65
accuracy			0.80	296
macro avg	0.85	0.56	0.55	296
weighted avg	0.82	0.80	0.74	296

Confusion Matrix:

```

[[230  1]
 [ 57  8]]

```

```

os.makedirs("../Reports", exist_ok=True)

with open("../Reports/evaluation.txt", "w") as f:
    f.write(f"Accuracy: {acc}\n\n")
    f.write("Classification Report:\n")
    f.write(cls)
    f.write("\n\nConfusion Matrix:\n")
    f.write(str(cm))

print("evaluation.txt berhasil dibuat!")

```

✓ 0.0s

evaluation.txt berhasil dibuat!

```

importances = pipeline.named_steps["model"].feature_importances_
encoded_cols = pipeline.named_steps["preprocess"].get_feature_names_out()

fi_df = pd.DataFrame({
    "fitur": encoded_cols,
    "importance": importances
})

fi_top20 = fi_df.sort_values(by="importance", ascending=False).head(20)
fi_top20.to_csv("../Reports/feature_importances_top20.csv", index=False)

fi_top20

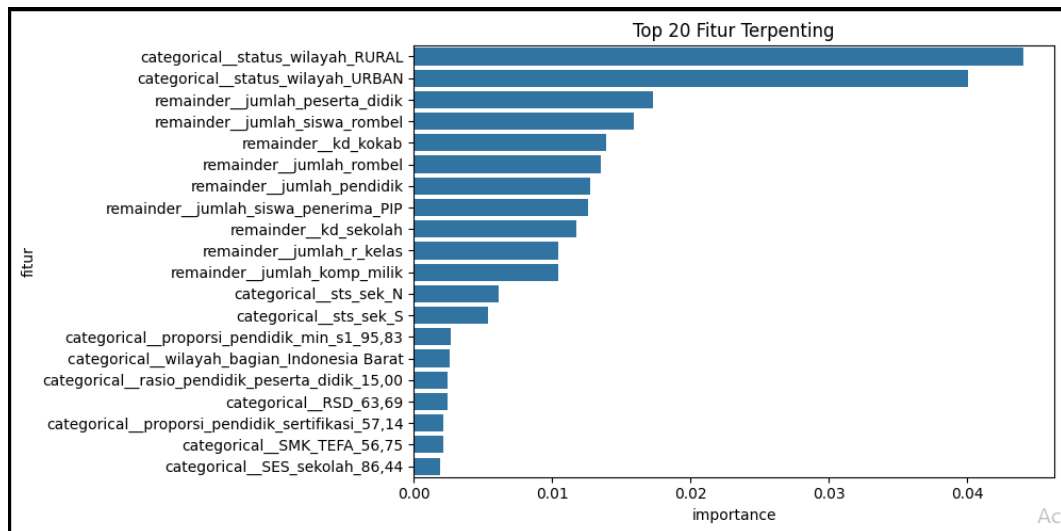
```

	fitur	importance
953	categorical_status_wilayah_RURAL	0.044082
954	categorical_status_wilayah_URBAN	0.040058
7067	remainder_jumlah_peserta_didik	0.017273
7073	remainder_jumlah_siswa_rombel	0.015874
7066	remainder_kd_kokab	0.013879
7072	remainder_jumlah_rombel	0.013529
7068	remainder_jumlah_pendidik	0.012759
7074	remainder_jumlah_siswa_penerima_PIP	0.012609
7065	remainder_kd_sekolah	0.011781
7069	remainder_jumlah_r_kelas	0.010466
7070	remainder_jumlah_komp_milik	0.010424
944	categorical_sts_sek_N	0.006164
945	categorical_sts_sek_S	0.005402
1009	categorical_proporsi_pendidik_min_s1_95,83	0.002696
950	categorical_wilayah_bagian_Indonesia Barat	0.002578
1514	categorical_rasio_pendidik_peserta_didik_15,00	0.002466
5782	categorical_RSD_63,69	0.002452
1254	categorical_proporsi_pendidik_sertifikasi_57,14	0.002157
6860	categorical_SMK_TEFA_56,75	0.002118
3599	categorical_SES_sekolah_86,44	0.001922

```

plt.figure(figsize=(10,6))
sns.barplot(data=fi_top20, x="importance", y="fitur")
plt.title("Top 20 Fitur Terpenting")
plt.tight_layout()
plt.savefig("../Reports/feature_importances_top20.png")
plt.show()

```



```
os.makedirs("../Models", exist_ok=True)

import pickle
with open("../Models/rf_pipeline_model.pkl", "wb") as f:
    pickle.dump({
        "pipeline": pipeline,
        "label_encoder": le,
        "numeric_cols": numeric_cols,
        "categorical_cols": categorical_cols
    }, f)

print("Model berhasil disimpan ke Models/rf_pipeline_model.pkl")
```

✓ 0.0s

Model berhasil disimpan ke Models/rf_pipeline_model.pkl

4. PROGRAM/deploy_gradio.py

```
1 # Deploy_Gradio.py
2 # Aplikasi Gradio untuk Prediksi Jenis Wilayah Sekolah (Urban / Rural)
3
4 import os
5 import pickle
6 import pandas as pd
7 import numpy as np
8 import gradio as gr
9
10 # =====
11 # Load Model
12 # =====
13 MODEL_PATH = "../Models/rf_pipeline_model.pkl"
14
15 if not os.path.exists(MODEL_PATH):
16     raise FileNotFoundError("Model belum ditemukan. Jalankan notebook terlebih dahulu!")
17
18 with open(MODEL_PATH, "rb") as f:
19     obj = pickle.load(f)
20
21 pipeline = obj["pipeline"]
22 le = obj["label_encoder"]
23 numeric_cols = obj["numeric_cols"]
24 categorical_cols = obj["categorical_cols"]
25
26 # =====
27 # UI Inputs (dipilih lebih ringkas)
28 # =====
29 ui_numeric = [
30     c for c in [
31         "jumlah_peserta_didik",
32         "jumlah_pendidik",
33         "proporsi_pendidik_min_sl",
34         "proporsi_pendidik_sertifikasi"
35     ] if c in numeric_cols
36 ]
37
38 ui_categorical = [
39     c for c in [
40         "kurikulum",
41         "jenis_sek",
42         "sts_sek",
43         "daerah_khusus",
44         "wilayah_bagian"
45     ] if c in categorical_cols
46 ]
47
48 inputs = []
49
50 for c in ui_numeric:
51     inputs.append(gr.Number(label=c))
52
53 for c in ui_categorical:
54     inputs.append(gr.Textbox(label=c))
55
56 # =====
57 # Mapping Admin = Urban/Rural
58 # =====
59 mapping = {
60     "KOTA": "URBAN",
61     "KABUPATEN": "RURAL"
62 }
63
64 # =====
65 # Prediction Function
66 # =====
67 def predict(vals):
68     data = {}
69
70     # isi kolom numerik
71     i = 0
72     for col in ui_numeric:
73         data[col] = [vals[i]]
74         i += 1
75
76     # isi kolom kategorikal
77     for col in ui_categorical:
78         data[col] = [vals[i]]
79         i += 1
80
81     # pastikan semua kolom terisi
82     for col in numeric_cols:
83         if col not in data:
84             data[col] = [0]
85
86     for col in categorical_cols:
87         if col not in data:
88             data[col] = [""]
89
90     # buat DataFrame untuk prediksi
91     X = pd.DataFrame(data)[numeric_cols + categorical_cols]
92
93     # prediksi
94     pred = pipeline.predict(X)[0]
95     proba = pipeline.predict_proba(X)[0]
96
97     # label asli dataset = "KOTA" / "KABUPATEN"
98     label_admin = le.inverse_transform([pred])[0]
99
100     # hasil mapping = "URBAN" / "RURAL"
101     final_label = mapping.get(label_admin, "UNKNOWN")
102
103     # probabilitas dimapping juga
104     proba_named = {}
105     for idx, p in enumerate(proba):
106         original_label = le.inverse_transform([idx])[0]
107         mapped_label = mapping.get(original_label, original_label)
108         proba_named[mapped_label] = float(p)
109
110     # output final
111     return f"({final_label}) ({label_admin})", proba_named
112
113 # =====
114 # Gradio App
115 # =====
116 outputs = [
117     gr.Label(label="Prediksi Jenis Wilayah"),
118     gr.Label(label="Probabilitas (Urban/Rural)")
119 ]
120
121 app = gr.Interface(
122     fn=predict,
123     inputs=inputs,
124     outputs=outputs,
125     title="Prediksi Jenis Wilayah Sekolah",
126     description="Aplikasi Machine Learning dengan Random Forest untuk menentukan jenis wilayah sekolah (URBAN/RURAL).",
127 )
128
129 # =====
130 # Jalankan Aplikasi
131 # =====
132 if __name__ == "__main__":
133     app.launch()
134
```

